Hardness of Submodular Cost Allocation: Lattice Matching and a Simplex Coloring Conjecture

Alina Ene[*1,2] **and Jan Vondrák**[3]

1 **Center for Computational Intractability, Princeton University**
aene@cs.princeton.edu
2 **Department of Computer Science and DIMAP, University of Warwick**
3 **IBM Almaden Research Center**
jvondrak@us.ibm.com

Abstract

We consider the Minimum Submodular Cost Allocation (MSCA) problem [3]. In this problem, we are given k submodular cost functions $f_1, \ldots, f_k : 2^V \to \mathbb{R}_+$ and the goal is to partition V into k sets $A_1, \ldots, A_k$ so as to minimize the total cost $\sum_{i=1}^{k} f_i(A_i)$. We show that MSCA is inapproximable within any multiplicative factor even in very restricted settings; prior to our work, only Set Cover hardness was known. In light of this negative result, we turn our attention to special cases of the problem. We consider the setting in which each function f_i satisfies $f_i = g_i + h$, where each g_i is monotone submodular and h is (possibly non-monotone) submodular. We give an $O(k \log |V|)$ approximation for this problem. We provide some evidence that a factor of k may be necessary, even in the special case of Hypergraph Labeling [3]. In particular, we formulate a simplex-coloring conjecture that implies a Unique-Games-hardness of $k - 1 - \epsilon$ for k-uniform Hypergraph Labeling and label set $[k]$. We provide a proof of the simplex-coloring conjecture for $k = 3$.[1]

1998 ACM Subject Classification Analysis of Algorithms and Problem Complexity, Optimization

Keywords and phrases Minimum Cost Submodular Allocation, Submodular Optimization, Hypergraph Labeling

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.144

1 Introduction

Labeling problems arise in a number of applications including document classification, image segmentation, facility location, and others. The general problem asks for a labeling of a ground set V by k labels in a way that minimizes a certain notion of "cost"; this cost can penalize "similar elements" being labeled differently, or elements being assigned a label that they "do not prefer". A classical example is the Graph Multiway Cut problem where given a graph $G = (V, E)$ with k terminals $t_1, \ldots, t_k \in V$, we want to partition the vertices into k disjoint sets $S_1, \ldots, S_k$ such that $t_i \in S_i$ and we minimize the number of edges between different parts. Over time, more general versions of this problem have been proposed in order to capture the fact that vertices might have more nuanced (weighted) preferences to be labeled in a certain way [16], relationships more general than pairwise might be present [11],

* Part of this work was done while the author was visiting IBM Almaden. Supported in part by NSF grant CCF-1016684 and a Chirag Foundation graduate fellowship.
[1] Subsequent to this work, a proof of the simplex-coloring conjecture has been found [17].

17th Int'l Workshop on Approximation Algorithms for Combinatorial Optimization Problems (APPROX'14) / 18th Int'l Workshop on Randomization and Computation (RANDOM'14).
Editors: Klaus Jansen, José Rolim, Nikhil Devanur, and Cristopher Moore; pp. 144–159

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

etc. This led to the study of problems such as Metric Labeling [16], 0-Extension [4], and Node-weighted / Hypergraph Multiway Cut [10]. The main object of our study is an abstract version of this problem, the Minimum Submodular Cost Allocation (MSCA) problem, introduced in [3]. In this problem, the cost function associated with each label is a submodular function; $f : 2^V \to \mathbb{R}$ is submodular if $f(A) + f(B) \geq f(A \cap B) + f(A \cup B)$ for every $A, B \subseteq V$.

Minimum Submodular Cost Allocation (MSCA). *Given k submodular functions $f_1, \ldots, f_k : 2^V \to \mathbb{R}_+$, find a partition $(A_1, \ldots, A_k)$ of V that minimizes $\sum_{i=1}^k f_i(A_i)$.*

This captures problems such as Graph Multiway Cut, Hypergraph Multiway Cut, and Uniform Metric Labeling due to the fact that the cut function in graphs and hypergraphs is submodular. However, MSCA is considerably more general than these problems. The special case of the problem in which each of the functions f_i is monotone is equivalent to the Submodular Facility Location problem considered by Svitkina and Tardos [24] who gave an $O(\log |V|)$-approximation and a matching hardness via a reduction from Set Cover. Following previous work, Chekuri and Ene [3, 2] considered several special cases of the MSCA problem in which the functions are non-monotone but they have additional structure. In [3] it was shown that, if each function f_i is the sum of a monotone submodular function g_i and a symmetric submodular function h that is the same for all of the labels, one can achieve an $O(\log |V|)$ approximation. Furthermore, several well-studied partitioning problems such as Graph Multiway Cut and Node-weighted Multiway Cut can be cast as special cases of MSCA in which the functions f_i arise from a single underlying submodular function. Submodular Multiway Partition, a special case of MSCA, was proposed as a unifying umbrella for these problems [25] and shown to admit a $(2 - 2/k)$-approximation which is best possible [2, 7].

Despite this progress, the MSCA problem itself was not very well understood. On the hardness side, prior to our work the best lower bound for the general MSCA problem was the Set Cover hardness shown by Svitkina and Tardos [24] for the special case in which all of the functions are monotone. On the positive side, no approximation guarantees have been known for the MSCA problem with non-monotone functions. This is perhaps surprising, since submodular minimization problems typically admit at least polynomial approximation factors; for example, $O(|V|)$ or $\tilde{O}(\sqrt{|V|})$ approximations are achievable for several minimization problems with submodular costs [14, 12, 23]. One of the main questions left open by previous work was to bridge this wide gap.

Our Results. In this paper, we show that the MSCA problem is in fact inapproximable within *any multiplicative factor* even in very restricted settings. The following theorem formally states our main hardness result.

▶ **Theorem 1.** *It is* **NP***-hard to decide whether the optimal value for a* Minimum Submodular Cost Allocation *problem is zero, even for $k = 3$ and functions of the form $f_i(S) = c_i(S) + \delta_{G_i}(S)$ where $c_i(S) = \sum_{j \in S} c_{ij}$ with 0/1 coefficients c_{ij} and δ_{G_i} is the cut function of a directed graph[2] G_i.*

As an intermediate result, we prove the NP-completeness of two related combinatorial problems that we call Lattice Matching and Partition Matching (see Section 2); we believe this might be of independent interest.

In light of this negative result, we turn our attention to special cases of the problem. In particular, we consider the following problem introduced in [3].

[2] The cut function of $G = (V, A)$ is $\delta_G(S) = |\{(v, w) \in A : v \in S, w \notin S\}|$.

Monotone-restricted MSCA. *For each $i \in [k]$, let $g_i : 2^V \to \mathbb{R}_+$ be a* monotone *submodular function (the "assignment cost). and let $h : 2^V \to \mathbb{R}_+$ be a (possibly non-monotone) submodular function (the "separation cost"). The* Monotone-restricted MSCA *problem is the special case of* MSCA *in which $f_i = g_i + h$ for each $i \in [k]$.*

As mentioned above, [3] gave an $O(\log |V|)$ approximation for a special case where the separation cost function h is *symmetric*. This is best possible even for $h = 0$ which yields the Submodular Facility Location problem [24]. In this paper, we give an $O(k \log |V|)$ approximation for the general Monotone-restricted MSCA problem in which the separation cost function h is not necessarily symmetric.

▶ **Theorem 2.** *There is an $O(k \log |V|)$-approximation for the* Monotone-restricted MSCA *problem.*

Our approach is based on a reduction to the symmetric case via symmetrization of the separation cost h. This result, although quite straightforward, provides us with a very general setting in which the MSCA problem admits a non-trivial approximation. The remaining question is whether the factor of k in Theorem 2 can be eliminated. We provide some evidence that the factor of k may be necessary for the following special case of the problem, introduced in [3].

Hypergraph Labeling. *Given a hypergraph $H = (V, E)$ with edge weights $w(e) \geq 0$ and vertex assignment costs $c(v, i) \geq 0$, find a labeling $\ell : V \to [k]$ so as to minimize $\sum_{v \in V} c(v, \ell(v)) + \sum_{e \in E : |\ell[e]| > 1} w(e)$.*

In other words, we want to minimize the assignment costs of the vertices plus the weight of the edges that are cut (receive multiple labels). This problem is a common generalization of Uniform Metric Labeling [16] and Hypergraph Multiway Cut [18]; i.e., instead of pairwise relationships we consider multi-tuple interactions and we also have modular assignment costs. On the other hand, Hypergraph Labeling can be cast as a special case of Monotone-restricted MSCA (see [3] or Section 4). Building on the work of Kleinberg and Tardos [16] for Uniform Metric Labeling, Chekuri and Ene [3] gave a d-approximation for the Hypergraph Labeling problem, where d is the maximum size of a hyperedge. Our result above gives an $O(k \log |V|)$-approximation for Hypergraph Labeling.

We provide some evidence that a factor of k might be necessary, in particular when $k = d$. We propose a conjecture (somewhat reminiscent of Sperner's Lemma [22]) which implies that a natural LP relaxation of the Hypergraph Labeling problem has integrality gap $k - 1$ for k-uniform hypergraphs and any approximation factor below $k - 1$ would refute the Unique Games Conjecture (using a general result of [7]). We prove the conjecture in the special case of $k = 3$ and thus we obtain an integrality gap and Unique Games hardness of $2 - \epsilon$ for this case; previously, only an integrality gap of $4/3$ was known [16].

Organization. The rest of the paper is organized as follows. In Section 2, we prove the inapproximability of the general MSCA problem, as well as the **NP**-completeness of the related Lattice Matching and Partition Matching problems. In Section 3, we show our approximation result for the Monotone-restricted MSCA problem. In Section 4, we discuss a conjecture that would imply hardness for the Hypergraph Labeling problem. Some details and proofs are deferred to the appendix.

2 Hardness of Minimum Submodular Cost Allocation

If $k = 2$, the MSCA problem can be reduced to submodular function minimization as follows. Let $f : 2^V \to \mathbb{R}_+$ be the function such that $f(S) = f_1(S) + f_2(V \setminus S)$. It is easy to see that f is submodular (Proposition 16). A submodular function can be minimized in polynomial time [5, 19, 21, 13, 15], and therefore MSCA is in **P** when $k = 2$.

The main result of this section is that for $k \geq 3$, the MSCA problem does not admit any finite approximation factor. In particular, we prove that for $k \geq 3$ it is **NP**-hard to decide whether the optimal solution has value zero or nonzero. We use functions in a particular form, using the cut function of a directed graph: $\delta_G(S) = |\{(v, w) \in A(G) : v \in S, w \notin S\}|$.

▶ **Theorem 3.** *It is* **NP***-hard to decide whether the optimal value for an instance of* MSCA *is zero, even for $k = 3$ and functions of the form $f_i(S) = c_i(S) + \delta_{G_i}(S)$ where c_i is a linear function and δ_{G_i} is the cut function of a directed graph G_i.*

We start with the following well-known fact. For any non-negative submodular function, the collection of sets of zero value forms a *Boolean lattice* (see Proposition 14; we recall that a Boolean lattice is a family of sets $\mathcal{L} \subseteq 2^V$ closed under unions and intersections, i.e. $\forall A, B \in \mathcal{L}; A \cap B \in \mathcal{L}, A \cup B \in \mathcal{L}$). This observation suggests that the question of checking for solutions of zero value is related to the following problem that we call Lattice Matching.

Lattice Matching. *Given k Boolean lattices $\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_k \subset 2^V$ (by a suitable compact representation), find k disjoint sets $S_1 \in \mathcal{L}_1, \ldots, S_k \in \mathcal{L}_k$ such that $\bigcup_{i=1}^k S_i = V$.*

We show that this problem is **NP**-complete for $k \geq 3$, by a reduction from 1-in-3 SAT [20]. A technical issue is the question of representing a lattice compactly on the input. For this purpose we use a representation of lattices by directed graphs that goes back to Birkhoff [1]. In fact this construction also provides explicit submodular functions whose zeros are exactly the points of the respective lattice, and hence we prove Theorem 3.

A special case of a lattice is the collection of all unions of sets from some partition (collection of disjoint sets) $\mathcal{P}_i$. Thus the Lattice Matching problem has the following natural special case.

Partition Matching. *Given k collections of sets $\mathcal{P}_1, \ldots, \mathcal{P}_k \subset 2^V$, where each $\mathcal{P}_i$ is a partition of a subset of V (the sets in $\mathcal{P}_i$ are disjoint), find disjoint sets $S_1, \ldots, S_\ell \in \bigcup_{i=1}^k \mathcal{P}_i$ such that $\bigcup_{i=1}^\ell S_i = V$.*

Just like the problems above, the Partition Matching problem is in **P** for $k \leq 2$. We prove that this problem is **NP**-complete for $k = 5$. We leave it as an open question whether it is **NP**-complete for $k = 3$ and $k = 4$.

2.1 Representation of Lattices by Directed Graphs

In the following, we describe how to encode a Boolean lattice using a directed graph, and how this connects MSCA and Lattice Matching. We follow the construction of [8].

▶ **Definition 4.** Given a lattice $\mathcal{L} \subseteq 2^V$ such that $\emptyset \in \mathcal{L}$, let $V_\mathcal{L} = \bigcup \{S : S \in \mathcal{L}\}$. For each $v \in V_\mathcal{L}$, let $D(v) = \bigcap \{S : S \in \mathcal{L}, v \in S\}$. We define a directed graph $G_\mathcal{L} = (V_\mathcal{L}, A)$ where $A = \{(v, w) : v \in V_\mathcal{L}, w \in D(v), w \neq v\}$.

The following lemma is implicit in [1, 8]. We include a simple proof for completeness.

▶ **Lemma 5.** *For every lattice $\mathcal{L} \subseteq 2^V$ such that $\emptyset \in \mathcal{L}$, the directed graph $G_\mathcal{L}$ encodes the lattice in the sense that $S \in \mathcal{L}$ if and only if $S \subseteq V_\mathcal{L}$ and $G_\mathcal{L}$ has no arcs from S to $V_\mathcal{L} \setminus S$.*

Proof. If $S \in \mathcal{L}$, then $S \subseteq V_\mathcal{L}$ by the properties of the lattice. Also, for each each $v \in S$, $D(v) = \bigcap\{S' \in \mathcal{L} : v \in S'\} \subseteq S$ and hence all arcs originating at v stay within S.

Conversely, if $S \subseteq V_\mathcal{L}$ and there are no arcs leaving S, we know that for each $v \in S$, $D(v) \subseteq S$. By the properties of lattices, $D(v) = \bigcap\{S' \in \mathcal{L}, v \in S'\} \in \mathcal{L}$. If $S \neq \emptyset$, we get that $S = \bigcup_{v \in S} D(v) \in \mathcal{L}$. If $S = \emptyset$, then $S \in \mathcal{L}$ by assumption. ◀

Thus the directed graph $G_\mathcal{L}$ encodes the lattice $\mathcal{L}$ in a compact way: its description has size $O(n^2)$, where $n = |V|$. Furthermore, we observe that this description provides a submodular function whose zeros are exactly the sets in $\mathcal{L}$.

▶ **Lemma 6.** *For a lattice $\mathcal{L} \subset 2^V$ defined by the directed graph $G_\mathcal{L}$, the following function is submodular and its zeros are exactly the sets in $\mathcal{L}$:*

$$f_\mathcal{L}(S) = |S \setminus V_\mathcal{L}| + \delta_{\mathcal{G}_\mathcal{L}}(S)$$

where $\delta_{\mathcal{G}_\mathcal{L}}(S) = |\{(v, w) \in A(\mathcal{G}_\mathcal{L}) : v \in S, w \notin S\}|$ is the directed cut function of $G_\mathcal{L}$.

Proof. By Lemma 5, $S \in \mathcal{L}$ if and only if $S \subseteq V_\mathcal{L}$ and there are no arcs from S to outside of S in $G_\mathcal{L}$, which occurs if and only if $f_\mathcal{L}(S) = 0$. ◀

It follows that the Lattice Matching problem – where the lattices are given by its associated directed graph – is equivalent to checking whether the MSCA instance in which the functions are $\{f_{\mathcal{L}_i} \mid i \in [k]\}$ has zero cost. To prove Theorem 3, it remains to prove the **NP**-completeness of Lattice Matching under this encoding.

2.2 NP-completeness of Partition Matching and Lattice Matching

In this section, we prove that the Lattice Matching problem is **NP**-complete for $k = 3$. First, as a warm-up, let us prove that its special case, the Partition Matching problem, is **NP**-complete for $k = 7$. We reduce from the following **NP**-complete problem [9].

3-bounded 3-set Packing. *Given a system of triples $\mathcal{T} \subseteq 2^V$ such that each element of V is contained in at most 3 triples, it is **NP**-complete to decide whether there exists a collection of disjoint triples covering V.*

▶ **Theorem 7.** Partition Matching *is **NP**-complete for $k = 7$.*

Proof. Given an instance of 3-bounded 3-set Packing, we observe that each triple intersects at most 6 other triples (2 for each of its elements). Thus we can inductively color the triples with 7 colors in such a way that intersecting triples get different colors. We define $\mathcal{P}_i$ to be the collection of all triples of color i. We obtain an instance of Partition Matching with $k = 7$, for which it is **NP**-complete to decide whether there exists a collection of disjoint triples covering V. ◀

For lower values of k, we use more careful reductions from the Monotone 1-in-3 SAT problem [20].

Monotone 1-in-3 SAT. *Given a formula* $\bigwedge_{i=1}^{m}(x_{i_1} \vee x_{i_2} \vee x_{i_3})$ *(without negations), it is* **NP***-complete to find a Boolean assignment such that in each clause* $(x_{i_1} \vee x_{i_2} \vee x_{i_3})$*, exactly one variable is True and two variables are False.*

▶ **Theorem 8.** Partition Matching *is* **NP***-complete for* $k = 5$.

Proof. Given an instance of Monotone 1-in-3 SAT, we produce an instance of Partition Matching as follows. We define a ground set V consisting of

- An element v_j for each variable x_j.
- Two elements $x_j^i, \neg x_j^i$ for each occurrence of a variable x_j in clause i.

On this ground set, we define the following 5 collections of sets:

- $\mathcal{P}_1$ contains for each variable x_j a set $\{v_j\} \cup \{x_j^i : \forall \text{clause } i \text{ containing variable } x_j\}$.
- $\mathcal{P}_2$ contains for each variable x_j a set $\{v_j\} \cup \{\neg x_j^i : \forall \text{clause } i \text{ containing variable } x_j\}$.
- $\mathcal{P}_3$ contains for each clause $x_{i_1} \vee x_{i_2} \vee x_{i_3}$ a set $\{x_{i_1}^i, \neg x_{i_2}^i, \neg x_{i_3}^i\}$.
- $\mathcal{P}_4$ contains for each clause $x_{i_1} \vee x_{i_2} \vee x_{i_3}$ a set $\{\neg x_{i_1}^i, x_{i_2}^i, \neg x_{i_3}^i\}$.
- $\mathcal{P}_5$ contains for each clause $x_{i_1} \vee x_{i_2} \vee x_{i_3}$ a set $\{\neg x_{i_1}^i, \neg x_{i_2}^i, x_{i_3}^i\}$.

We call the sets in $\mathcal{P}_1 \cup \mathcal{P}_2$ *variable-assignment* sets, and the sets in $\mathcal{P}_3 \cup \mathcal{P}_4 \cup \mathcal{P}_5$ *clause-assignment* sets. Observe that in each collection $\mathcal{P}_i$, the sets are pairwise disjoint. I.e., we have an instance of Partition Matching with $k = 5$.

If there is a Boolean assignment such that exactly one variable is each clause is satisfied, we produce a solution of Partition Matching as follows. For each variable $x_j =$ True, we choose the variable-assignment set containing v_j that is in $\mathcal{P}_2$ (i.e. containing all the elements $\neg x_j^i$). For each variable $x_j =$ False, we choose the variable-assignment set containing v_j that is in $\mathcal{P}_1$ (i.e. containing all the elements x_j^i). Finally, for each clause $x_{i_1} \vee x_{i_2} \vee x_{i_3}$, we choose the set corresponding to its assignment, from either $\mathcal{P}_3$, $\mathcal{P}_4$ or $\mathcal{P}_5$. It is easy to verify that these sets are disjoint and cover the entire ground set V.

Conversely, let us assume that there is a collection of disjoint sets $\mathcal{F} \subset \mathcal{P}_1 \cup \mathcal{P}_2 \cup \mathcal{P}_3 \cup \mathcal{P}_4 \cup \mathcal{P}_5$ that covers the ground set V. Since $\mathcal{F}$ must cover the element v_j for each variable, it must contain a variable-assignment set in either $\mathcal{P}_1$ or $\mathcal{P}_2$ (no other sets contain v_j). This choice determines a Boolean assignment: we set $x_j =$ True if v_j is covered by a set from $\mathcal{P}_2$, and $x_j =$ False if v_j is covered by a set from $\mathcal{P}_1$. Now, consider the 6 elements $x_{i_1}^i, \neg x_{i_1}^i, x_{i_2}^i, \neg x_{i_2}^i, x_{i_3}^i, \neg x_{i_3}^i$ for clause i. Exactly 3 of these elements are covered by sets from $\mathcal{P}_1$ and $\mathcal{P}_2$, hence the remaining 3 elements must form a clause-assignment set in $\mathcal{P}_3 \cup \mathcal{P}_4 \cup \mathcal{P}_5$. These clause-assignment sets correspond to satisfying assignments and hence the formula is satisfied by the Boolean assignment that we defined. ◀

Finally, we prove that Lattice Matching is NP-complete for $k = 3$. Note that here we use the full flexibility of the Lattice Matching problem; we do not know whether the Partition Matching problem is NP-complete for $k = 3$.

▶ **Theorem 9.** Lattice Matching *is* **NP***-complete for* $k = 3$.

Proof. We use the same reduction as in the proof of Theorem 8, but we combine the 5 partitions into 3 lattices. Specifically, using the notation from the proof above, we define

- $\mathcal{L}_1 = \mathrm{cl}(\mathcal{P}_1 \cup \mathcal{P}_3)$
- $\mathcal{L}_2 = \mathrm{cl}(\mathcal{P}_2)$
- $\mathcal{L}_3 = \mathrm{cl}(\mathcal{P}_4 \cup \mathcal{P}_5)$

where $\mathrm{cl}(\mathcal{P})$ means all the sets that can be generated from $\mathcal{P}$ by taking unions and intersections. By construction, $\mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3$ are lattices that contain all the sets that were contained in $\mathcal{P}_1, \ldots, \mathcal{P}_5$ (as well as some additional sets). In the case of a satisfiable formula, we can still

choose disjoint sets covering V as above. Let S_i be the union of those of these sets that are contained in $\mathcal{L}_i$ (i.e., S_i is also in $\mathcal{L}_i$), and $S_1 \cup S_2 \cup S_3 = V$ is a feasible solution of the Lattice Matching problem. The potential issue with this construction is that we might have created a feasible solution of the Lattice Matching problem in case the formula is not satisfiable. Let us argue that this is not the case.

For each variable x_j, the element v_j is contained only in the variable-assignment sets arising from $\mathcal{P}_1$ and $\mathcal{P}_2$, and sets formed by unions of these variable-assignment sets with other sets in $\mathcal{L}_1 = \mathrm{cl}(\mathcal{P}_1 \cup \mathcal{P}_3), \mathcal{L}_2 = \mathrm{cl}(\mathcal{P}_2)$. Recall that these two variable assignment sets appear in $\mathcal{P}_1, \mathcal{P}_2$ respectively, and so their union/intersection is not part of the lattices that we generate. Therefore, v_j must be covered by a set that was generated from one of the two variable-assignment sets for x_j; depending on which one is used, we set x_j to True (if the set $\{v_j, \neg x_j^i, \neg x_j^{i'}, \ldots\}$ is used) or False (if the set $\{v_j, x_j^i, x_j^{i'}, \ldots\}$ is used).

By our construction of the three lattices, for each clause $x_{i_1} \vee x_{i_2} \vee x_{i_3}$, some of the respective elements $x_{i_1}^i, x_{i_2}^i, x_{i_3}^i, \neg x_{i_1}^i, \neg x_{i_2}^i, \neg x_{i_3}^i$ are going to appear as singletons in a lattice. In $\mathcal{L}_1$, we get new sets obtained by intersecting sets in $\mathcal{P}_1$ and $\mathcal{P}_3$: Specifically, this is the singleton $\{x_{i_1}^i\}$ for each clause $x_{i_1} \vee x_{i_2} \vee x_{i_3}$ (and sets obtained by taking unions of these singletons with other sets in $\mathcal{P}_1 \cup \mathcal{P}_3$). In $\mathcal{L}_2$, we obtain only the unions of sets in $\mathcal{P}_2$ (which are disjoint). In $\mathcal{L}_3$, we obtain by intersection the singleton $\{\neg x_{i_1}^i\}$ for each clause $x_{i_1} \vee x_{i_2} \vee x_{i_3}$, and again sets obtained by unions with other sets in $\mathcal{P}_4 \cup \mathcal{P}_5$ (recall the definitions of $\mathcal{P}_1, \mathcal{P}_2, \mathcal{P}_3, \mathcal{P}_4$ and $\mathcal{P}_5$).

Observe that the construction is not symmetric with respect to the three elements of a clause such as $x_{i_1}^i \vee x_{i_2}^i \vee x_{i_3}^i$. While $\{x_{i_1}^i\}$ and $\{\neg x_{i_1}^i\}$ appear as sets in $\mathcal{L}_1, \mathcal{L}_3$ respectively, $\{x_{i_2}^i\}$ and $\{\neg x_{i_2}^i\}$ do not appear as singletons in any lattice. This is because $x_{i_2}^i$ appears in exactly one set in $\mathcal{P}_4$, and $\neg x_{i_2}^i$ appears in exactly one set in $\mathcal{P}_3$ and one set in $\mathcal{P}_5$. Thus we do not form any intersections that can produce $\{x_{i_2}^i\}$ or $\{\neg x_{i_2}^i\}$. By the same argument, we do not form any intersections that can produce a pair containing $x_{i_2}^i$ or $\neg x_{i_2}^i$. Every set in $\mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3$ that contains $x_{i_2}^i$ or $\neg x_{i_2}^i$ contains either the variable-assignment set $\{v_j, \ldots, \}$, or a triple corresponding to a satisfying assignment of the i-th clause, e.g. $\{x_{i_1}^i, \neg x_{i_2}^i, \neg x_{i_3}^i\}$. If x_{i_2} was set to True, then $\neg x_{i_2}^i$ is covered by a variable-assignment set but $x_{i_2}^i$ is not because that would cause v_j to be covered twice. Therefore, the only way that $x_{i_2}^i$ can be covered is by a triple corresponding to a satisfying assignment of the i-th clause. By the same argument, this assignment is consistent with our assignment of variables. We can repeat this argument for each clause; it proves that if there is a feasible solution of the Lattice Matching problem, then our assignment satisfies every clause of the formula. ◀

3 Monotone-restricted MSCA Algorithm

In the previous section, we showed that the MSCA problem does not admit any multiplicative approximation whatsoever. This can be viewed as evidence that MSCA is "not the right generalization" of problems like Multiway Cut, Uniform Metric Labeling, etc. In this section we restrict MSCA to some extent, so that we still obtain a fairly general partitioning problem but one that allows some non-trivial approximation. We consider the Monotone-restricted MSCA problem in which each assignment cost function g_i is an arbitrary monotone submodular function and the separation cost function h is an arbitrary submodular function (but the same one for all label values). We seek a partitioning (or labeling) $(S_1, S_2, \ldots, S_k)$ minimizing $\sum_{i=1}^{k}(g_i(S_i) + h(S_i))$.

We observe that, for any submodular function h, $h'(S) = h(S) + h(V \setminus S)$ is a symmetric submodular function (see Proposition 17 in the appendix). Since h' is symmetric, we can use

the algorithm of [3] to construct a labeling for the instance of Monotone-restricted MSCA in which the assignment costs are given by g_i and the separation cost is given by h'. For any labeling, we can relate its h' cost to its h cost as follows.

▶ **Proposition 10.** *Let $(A_1, \ldots, A_k)$ be a labeling. We have*

$$\sum_{i=1}^{k} h(A_i) \leq \sum_{i=1}^{k} h'(A_i) \leq k \sum_{i=1}^{k} h(A_i).$$

Proof. The first inequality follows from the fact that h is non-negative. Therefore it suffices to show the second inequality. A non-negative submodular function is sub-additive and thus we have

$$h(V \setminus A_i) = h(\cup_{j \neq i} A_j) \leq \sum_{j \neq i} h(A_j).$$

Therefore $\sum_{i=1}^{k} h(V \setminus A_i) \leq (k-1) \sum_{i=1}^{k} h(A_i)$ and $\sum_{i=1}^{k} h'(A_i) \leq k \sum_{i=1}^{k} h(A_i)$. ◀

Let OPT and OPT$'$ be the costs of the optimal solution for the original instance in which the separation cost function is h and the modified instance in which the separation cost function is h', respectively. By the above, OPT$' \leq k \cdot$ OPT. Let $(A_1, \ldots, A_k)$ be the solution constructed by the algorithm of [3] for the modified instance. The result of [3] is that there is an $O(\log |V|)$-approximation for Monotone-restricted MSCA whenever the functions g_i are monotone submodular and h is symmetric submodular. It follows that

$$\sum_{i=1}^{k} g_i(A_i) + \sum_{i=1}^{k} h(A_i) \leq \sum_{i=1}^{k} g_i(A_i) + \sum_{i=1}^{k} h'(A_i) \leq O(\log |V|)\text{OPT}' \leq O(k \log |V|)\text{OPT}.$$

Therefore we have the following theorem.

▶ **Theorem 11.** *There is an $O(k \log |V|)$-approximation for the* Monotone-restricted MSCA *problem.*

We remark that the factor of $\log |V|$ is necessary due to the hardness of the Submodular Facility Location problem, which is the case of $h = 0$. The same hardness can be obtained when h is a simple symmetric submodular function and the g_i's are modular functions – see Appendix B.

4 Hypergraph Labeling and Sperner's Colorings

In this section, we consider the Hypergraph Labeling problem, which is a special case of Monotone-restricted MSCA (see Section 1 for a definition). As we have shown in the previous section, Monotone-restricted MSCA (and thus Hypergraph Labeling) admits an $O(k \log |V|)$ approximation, where k is the number of labels. Also, the Hypergraph Labeling problem was shown to admit a d-approximation when the size of each hyperedge is at most d [3]. We provide some evidence that a factor of k might be necessary for this problem, in particular when $d = k$.

4.1 LP Relaxations for Hypergraph Labeling

Chekuri and Ene [3] gave a convex-programming relaxation (LE-Rel) for MSCA that is based on the Lovász extension of a submodular function. Let us review the convex relaxation LE-Rel in the special case of the Hypergraph Labeling problem. In LE-Rel, we have variables $x_{v,i}$ for

$v \in V, i \in [k]$. Recall that the objective function in Hypergraph Labeling can be modeled as $\sum_{i=1}^{k} f_i(S_i)$ where $f_i(S) = g_i(S) + h(S)$, $g_i(S) = \sum_{v \in S} c(v,i)$ and

$$h(S) = \sum_{e \in E : r(e) \in S, e \not\subseteq S} w(e)$$

is the rooted version of the hypergraph cut function, for some choice of a root $r(e) \in e$ for every $e \in E$.

The LE-Rel relaxation is based on the Lovász extension of the objective functions $f_i(S)$. By linearity, the Lovász extension $\hat{f}_i(\mathbf{x})$ can be written as $\hat{f}_i(\mathbf{x}) = \hat{g}_i(\mathbf{x}) + \hat{h}(\mathbf{x}) = \sum_{v \in V} c(v,i) x_{v,i} + \hat{h}(\mathbf{x})$ since the function g_i is linear. The Lovász extension of the rooted hypergraph cut function h can be written as follows: for a uniformly random threshold $\lambda \in [0,1]$,

$$\hat{h}(\mathbf{x}) = \sum_{e \in E} w(e) \Pr[x_{r(e)} > \lambda \ \& \ \exists v \in e; x_v \leq \lambda] = \sum_{e \in E} w(e)(x_{r(e)} - \min_{v \in e} x_v)$$

(see [6] for details). The objective function of LE-Rel is $\sum_{i=1}^{k} \hat{f}_i(\mathbf{x}_i) = \sum_{i=1}^{k} (\hat{g}_i(\mathbf{x}_i) + \hat{h}(\mathbf{x}_i))$, where $\mathbf{x}_i \in \mathbb{R}^V$ is the assignment vector for label i. We note that $\sum_{i=1}^{k} x_{r(e),i} = 1$ by the assignment constraint in LE-Rel, and hence we have

$$\sum_{i=1}^{k} \hat{h}(\mathbf{x}_i) = \sum_{e \in E} w(e) \left(1 - \sum_{i=1}^{k} \min_{v \in e} x_{v,i}\right).$$

Hence we can write the full LE-Rel relaxation for Hypergraph Labeling as follows.

LE-Rel for Hypergraph Labeling

$$\min \quad \sum_{v \in V} \sum_{i=1}^{k} c(v,i) x_{v,i} + \sum_{e \in E} w(e) \left(1 - \sum_{i=1}^{k} \min_{v \in e} x_{v,i}\right) :$$

$$\sum_{i=1}^{k} x_{v,i} = 1 \qquad \forall v \in V$$

$$x_{v,i} \geq 0 \qquad \forall v \in V, i \in [k]$$

Formally, this is not in the form of a linear program but it is easy to see that the expression $\min_{v \in e} x_{v,i}$ can be replaced by a new variable $z_{e,i}$ with constraints $z_{e,i} \leq x_{v,i} \forall v \in e$. We prefer to keep the form above for compactness.

Next, we observe that this LP is equivalent to the "Local Distribution LP" considered in [7]. In the Local Distribution LP, we have $x_{v,i}$ variables as above, and also $y_{e,\alpha}$ variables for each hyperedge $e \in E$ and each possible assignment $\alpha \in [k]^e$. The hyperedge variables $y_{e,\alpha}$ can be interpreted as a distribution over labelings of the respective hyperedge e. The hyperedge variables must be consistent with the vertex variables in the sense that all assignments such that $\alpha_v = i$ should add up to $\sum_{\alpha \in [k]^e : \alpha_v = i} y_{e,\alpha} = x_{v,i}$. The Local Distribution LP reads as follows.

Local Distribution LP for Hypergraph Labeling

$$\min \quad \sum_{v\in V}\sum_{i=1}^{k} c(v,i)x_{v,i} + \sum_{e\in E,\alpha\neq(\ell,\ell,\ldots,\ell)} w(e)y_{e,\alpha} :$$

$$\sum_{\alpha\in[k]^e,\alpha_v=i} y_{e,\alpha} = x_{v,i} \qquad \forall v\in e\in E, i\in[k]$$

$$\sum_{i=1}^{k} x_{v,i} = 1 \qquad \forall v\in V$$

$$x_{v,i}, y_{e,\alpha} \geq 0 \qquad \forall v,i,e,\alpha$$

Consider a feasible assignment to the variables $x_{v,i}$. Given this assignment, the Local Distribution LP aims to minimize the cut cost $\sum_{\alpha\neq(\ell,\ell,\ldots,\ell)} y_{e,\alpha}$ for each hyperedge e, subject to the condition $\sum_{\alpha\in[k]^e;\alpha_v=i} y_{e,\alpha} = x_{v,i}$. We claim that the optimal way to do this is to set $y_{e,(i,i,\ldots,i)} = \min_{v\in e} x_{v,i}$ for each $i\in[k]$, and then distribute the remaining mass $1-\sum_{i=1}^{k}\min_{v\in e} x_{v,i}$ among variables $y_{e,\alpha}$ where α contains more than 1 label, so as to satisfy the consistency constraints $\sum_{\alpha\in[k]^e,\alpha_v=i} y_{e,\alpha} = x_{v,i}$. This is possible to do greedily, since as long as we have $\sum_{\alpha\in[k]^e} y_{e,\alpha} < 1$, there is some label for each vertex such that $\sum_{\alpha\in[k]^e,\alpha_v=i} y_{e,\alpha} < x_{v,i}$, and so we can increase the variable $y_{e,\alpha}$ for the corresponding assignment. This achieves the objective value of $\sum_{v\in V}\sum_{i=1}^{k} c(v,i)x_{v,i} + \sum_{e\in E} w(e)(1-\sum_{i=1}^{k}\min_{v\in e} x_{v,i})$, identical to that of LE-Rel. On the other hand, $\min_{v\in e} x_{v,i}$ is the maximum value that we can assign to $y_{e,(i,i,\ldots,i)}$ without violating the consistency constraints, so the contribution of hyperedge e cannot be lower than $1-\sum_{i=1}^{k}\min_{v\in e} x_{v,i}$, just like in LE-Rel.

The work of [7] implies that for any variant of Min CSP including the Not-Equal predicate (which is the case here), it is Unique-Games-hard to achieve any approximation better than the integrality gap of the Local Distribution LP. Therefore, the LP presented here (in either equivalent form) is in some sense the optimal tool to consider when developing approximation algorithms for the Hypergraph Labeling problem.

4.2 A Simplex Coloring Conjecture

In this section, we describe a conjecture that would imply an integrality gap close to $k-1$ for the k-uniform Hypergraph Labeling problem with label set $[k]$.

Sperner's Simplex example. Let $q\geq 1$ be an integer and consider the $(k-1)$-dimensional simplex defined by

$$\Delta = \left\{\mathbf{x}=(x_1,x_2,\ldots,x_k)\in\mathbb{R}^k : \mathbf{x}\geq 0, \sum_{i=1}^{k} x_i = q\right\}.$$

We consider a vertex set of all the points in Δ with integer coordinates:

$$V = \left\{\mathbf{a}=(a_1,a_2,\ldots,a_k)\in\mathbb{Z}^k : \mathbf{a}\geq 0, \sum_{i=1}^{k} a_i = q\right\}.$$

We define an (unweighted) k-uniform hypergraph $H=(V,E)$ on this vertex set whose hyperedges are indexed by $\mathbf{b}\in\mathbb{Z}_+^k$ such that $\sum_{i=1}^{k} b_i = q-1$: we have

$$E = \left\{e(\mathbf{b}) : \mathbf{b}=(b_1,b_2,\ldots,b_k)\in\mathbb{Z}^k, \mathbf{b}\geq 0, \sum_{i=1}^{k} b_i = q-1\right\}$$

where

$$e(\mathbf{b}) = \{(b_1+1, b_2, \ldots, b_k), (b_1, b_2+1, \ldots, b_k), \ldots, (b_1, b_2, \ldots, b_k+1)\}.$$

For each vertex $\mathbf{a} \in V$, we have a list of admissible labels $L(\mathbf{a})$, which is

$$L(\mathbf{a}) = \{i \in [k] : a_i > 0\}.$$

Formally, in the setting of the Hypergraph Labeling problem, we define the assignment cost to be $c(\mathbf{a}, i) = 0$ whenever $i \in L(\mathbf{a})$ and $c(\mathbf{a}, i) = \infty$ otherwise. We also define the edge weights to be $w(e) = 1$ for all $e \in E$.

We call a labeling $\ell : V \to [k]$ Sperner-admissible if $\ell(\mathbf{a}) \in L(\mathbf{a})$ for each $\mathbf{a} \in V$. The reader may notice that this is a restriction identical to the framework of Sperner's Lemma [22], where the points on each lower-dimensional face can be labeled only with colors corresponding to vertices of that face. The conclusion of Sperner's Lemma is that there must exist a cell (a scaled copy of the simplex Δ) whose vertices have all k colors. We remark that this cell might not be a member of E since E consists only of scaled copies of Δ *without* rotation. Nevertheless, we need a different statement here.

▶ **Conjecture 12.** *For any Sperner-admissible labeling $\ell : V \to [k]$, there are at least $\binom{q+k-3}{k-2}$ hyperedges $e \in E$ that are not monochromatic under ℓ.*

Let us comment on where the expression $\binom{q+k-3}{k-2}$ comes from. The total number of hyperedges in E is the number of partitions of $q-1$ into a sum of k nonnegative integers. By a well-known combinatorial argument, this is equal to the number of choices of $k-1$ barriers from among $(q-1)+(k-1)$ points and barriers, which is $|E| = \binom{q+k-2}{k-1}$. Similarly, the number of hyperedges that are adjacent to a given facet of the simplex (e.g. those satisfying $b_k = 0$) is equal to the number of hyperedges in a similarly defined $(k-2)$-dimensional simplex, which is $\binom{q+k-3}{k-2}$. We conjecture that the labeling minimizing the number of non-monochromatic hyperedges is one that labels all vertices $\mathbf{a}$ with $a_1 > 0$ by label 1, and then it labels all vertices with $a_1 = 0$ arbitrarily (subject to the restrictions given above). Under this labeling, all the hyperedges $e(\mathbf{b})$ such that $b_1 > 0$ are labeled monochromatically by 1. The only hyperedges that receive more than 1 label are those where $b_1 = 0$, and the number of such hyperedges is $\binom{q+k-3}{k-2}$ as we argued above.

Implications for the Integrality Gap. Let us see what this conjecture would imply for the Hypergraph Labeling problem. We can view the geometric description above naturally as an LP solution for the Hypergraph Labeling problem where vertex $\mathbf{a}$ is mapped to $\mathbf{x_a} = \frac{1}{q}\mathbf{a}$. By construction, for each point $\mathbf{x_a}$ the nonzero coordinates are exactly those in the admissible list $L(\mathbf{a})$, so the assignment cost of this LP solution is $\sum_{v \in V} c(v,i)x_{v,i} = 0$. To compute the cut cost, consider a single hyperedge $e(\mathbf{b})$. The contribution of this hyperedge is

$$1 - \sum_{i=1}^{k} \min_{v \in e(\mathbf{b})} x_{v,i} = 1 - \sum_{i=1}^{k} \frac{1}{q} b_i = \frac{1}{q}$$

since we have $\sum_{i=1}^{k} b_i = q-1$ for every hyperedge $e(\mathbf{b})$. Thus, each hyperedge contributes $\frac{1}{q}$ to the LP cost and in total we have

$$LP = \frac{1}{q}|E| = \frac{1}{q}\binom{q+k-2}{k-1}.$$

In contrast, Conjecture 12 implies that for any feasible solution, at least $\binom{q+k-3}{k-2}$ hyperedges must be non-monochromatic and hence $OPT \geq \binom{q+k-3}{k-2}$. For $q \to \infty$, we obtain

$$\frac{OPT}{LP} = \frac{\binom{q+k-3}{k-2}}{\frac{1}{q}\binom{q+k-2}{k-1}} = \frac{q}{q+k-2}(k-1) \to k-1.$$

So Conjecture 12 implies that the integrality gap can be arbitrarily close to $k-1$ and consequently any approximation algorithm with a factor below $k-1$ would refute the Unique Games Conjecture [7].

4.3 Proof of Conjecture 12 for $k = 3$

▶ **Proposition 13.** *In the example above for $k = 3$ and $q \geq 1$, for any Sperner-admissible labeling $\ell : V \to [3]$ there are at least q triangles in E that are not monochromatic.*

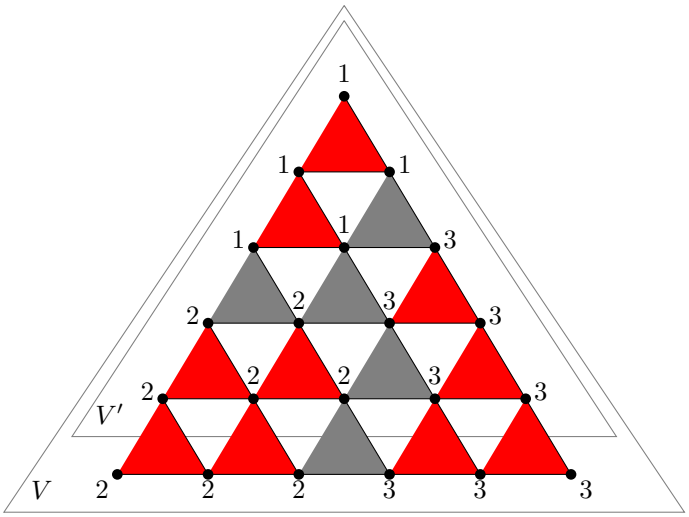

Figure 1 A Sperner-admissible coloring for $k = 3$ and $q = 5$. The set E of hyperedges consists of the shaded triangles. At least $q = 5$ triangles must be non-monochromatic (in gray). The sets V and V' are as in the proof.

Proof. We prove this statement by induction on q. If $q = 1$, then we have only one hyperedge in E which is required to be labeled $(1, 2, 3)$, so the statement holds.

If $q > 1$, let $V' = V \setminus \{\mathbf{a} \in V : a_1 = 0\}$ and $E' = E \cap \binom{V'}{3}$. Observe that $H' = (V', E')$ is the same hypergraph that our construction gives for $q - 1$, but the labeling restrictions are somewhat different. In particular, in V the bottom row (vertices with $a_1 = 0$) is allowed to be labeled only by colors $\{2, 3\}$, while in V' the bottom row (vertices with $a_1 = 1$) can be labeled by all 3 colors.

Consider any labeling $\ell : V \to [3]$. We distinguish two cases. If there is no vertex $\mathbf{a} \in V'$ such that $a_1 = 1$ and $\ell(\mathbf{a}) = 1$, then the labeling on V' is Sperner-admissible and we can apply induction to $H' = (V', E')$. The inductive hypothesis says that there are at least $q - 1$ non-monochromatic triangles in E'. Moreover, the bottom row of V is colored $\{2, 3\}$ and its endpoints have different colors. Therefore, there is an edge in the bottom row which is labeled $\{2, 3\}$. This gives 1 additional non-monochromatic triangle in $E \setminus E'$, for a total of q non-monochromatic triangles in E.

The remaining case is that there are some vertices in the bottom row of V' ($a_1 = 1$), labeled $\ell(\mathbf{a}) = 1$. Let the number of such vertices be s. Let us define a modified coloring $\ell' : V' \to \{2, 3\}$ where

- $\ell'(\mathbf{a}) = \ell(\mathbf{a})$ if $a_1 > 1$ or $\ell(\mathbf{a}) \neq 1$,
- $\ell'(\mathbf{a}) = 2$ if $\ell(\mathbf{a}) = 1$, $a_1 = 1$ and $a_2 > 0$,
- $\ell'(\mathbf{a}) = 3$ if $\ell(\mathbf{a}) = 1$, $a_1 = 1$ and $a_2 = 0$ (which implies $a_3 = q - a_1 - a_2 > 0$).

To summarize, starting from ℓ, we changed s labels in the bottom row of V' from 1 to 2 or 3. How many non-monochromatic triangles could we have added this way? The only new non-monochromatic triangles under ℓ' are those that were labeled $(1, 1, 1)$ under ℓ and they were adjacent to the bottom row. There could have been at most $s - 1$ such triangles, because each of them contains two vertices of the bottom row and the leftmost and rightmost vertex labeled 1 can appear only once in such a triangle. Therefore, we added at most $s - 1$ non-monochromatic triangles under ℓ' compared to ℓ.

Now, ℓ' is a Sperner-admissible labeling of $H' = (V', E')$. By the inductive hypothesis, there are at least $q - 1$ non-monochromatic triangles in E' under ℓ'. Consequently, there are at least $(q - 1) - (s - 1) = q - s$ non-monochromatic triangles in E' under the labeling ℓ. In addition, there are s non-monochromatic triangles in $E \setminus E'$, since every vertex labeled 1 in the bottom row of V' contributes one such triangle (its neighbors in the bottom row of V cannot be labeled 1). Therefore, there are at least q non-monochromatic triangles in E under ℓ. ◀

References

1 G. Birkhoff. Rings of sets. *Duke Mathematical Journal*, 3:443–454, 1937.

2 C. Chekuri and A. Ene. Approximation algorithms for submodular multiway partition. In *Proc. of IEEE FOCS*, 2011.

3 C. Chekuri and A. Ene. Submodular cost allocation problem and applications. In *Proc. of ICALP*, pages 354–366, 2011.

4 G. Călinescu, H. J. Karloff, and Y. Rabani. Approximation algorithms for the 0-extension problem. In *Proc. of ACM-SIAM SODA*, 2001.

5 W. H. Cunningham. On submodular function minimization. *Combinatorica*, 5(3):185–192, 1985.

6 A. Ene. Approximation algorithms for submodular optimization and graph problems. Ph.D. thesis, University of Illinois, Urbana-Champaign, 2013.

7 A. Ene, J. Vondrák, and Y. Wu. Local distribution and the symmetry gap: Approximability of multiway partitioning problems. In *Proc. of ACM-SIAM SODA*, pages 306–325, 2013.

8 S. Fujishige. *Submodular functions and optimization.* Elsevier Science, 2005.

9 M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman and Company, 1979.

10 N. Garg, V. V. Vazirani, and M. Yannakakis. Multiway cuts in node weighted graphs. *Journal of Algorithms*, 50(1):49–61, 2004.

11 D. Gibson, J. Kleinberg, and P. Raghavan. Clustering categorical data: An approach based on dynamical systems. *VLDB Journal*, 8(3-4):222–236, 2000.

12 G. Goel, C. Karande, P. Tripathi, and L. Wang. Approximability of combinatorial problems with multi-agent submodular cost functions. In *Proc. of IEEE FOCS*, pages 755–764, 2009.

13 S. Iwata, L. Fleischer, and S. Fujishige. A combinatorial, strongly polynomial-time algorithm for minimizing submodular functions. In *Proc. of ACM STOC*, pages 97–106, 2000.

14 S. Iwata and K. Nagano. Submodular function minimization under covering constraints. In *Proc. of IEEE FOCS*, pages 671–680, 2009.

15 S. Iwata and J. B. Orlin. A simple combinatorial algorithm for submodular function minimization. In *Proc. of ACM-SIAM SODA*, pages 1230–1237, 2009.

16 J. M. Kleinberg and E. Tardos. Approximation algorithms for classification problems with pairwise relationships: Metric labeling and Markov random fields. *Journal of the ACM*, 49(5):616–639, 2002.

17 M. Mirzakhani, 2014. personal communication.

18 K. Okumoto, T. Fukunaga, and H. Nagamochi. Divide-and-conquer algorithms for partitioning hypergraphs and submodular systems. *Algorithmica*, pages 1–20, 2010.

19 M. Queyranne. A combinatorial algorithm for minimizing symmetric submodular functions. In *Proc. of ACM-SIAM SODA*, pages 98–101, 1995.

20 T. J. Schaefer. The complexity of satisfiability problems. In *Proc. of ACM STOC*, pages 216–226, 1978.

21 A. Schrijver. A combinatorial algorithm minimizing submodular functions in strongly polynomial time. *Journal of Combinatorial Theory, Series B*, 80(2):346–355, 2000.

22 E. Sperner. Neuer Beweis für die Invarianz der Dimensionszahl und des Gebietes. *Math. Sem. Univ. Hamburg*, 6:265–272, 1928.

23 Z. Svitkina and L. Fleischer. Submodular approximation: Sampling-based algorithms and lower bounds. In *Proc. of IEEE FOCS*, pages 697–706, 2008.

24 Z. Svitkina and E. Tardos. Facility location with hierarchical facility costs. *ACM Transactions on Algorithms*, 6(2):1–22, 2010.

25 L. Zhao, H. Nagamochi, and T. Ibaraki. Greedy splitting algorithms for approximating multiway partition problems. *Mathematical Programming*, 102(1):167–183, 2005.

A Basic Lemmas

The following lemmas are folklore. We include the proofs for completeness.

▶ **Proposition 14.** *For every non-negative submodular function $f : 2^V \to \mathbb{R}_+$, the set $\mathcal{L} = \{S \subseteq V : f(S) = 0\}$ forms a lattice, i.e., it is closed under unions and intersections.*

Proof. Let $f(A) = f(B) = 0$. Then $f(A \cup B) + f(A \cap B) \leq f(A) + f(B) = 0$, and f is nonnegative, so we must have $f(A \cup B) = f(A \cap B) = 0$ as well. ◀

▶ **Proposition 15.** *Let $f : 2^V \to \mathbb{R}$ be a submodular function. Let $g : 2^V \to \mathbb{R}$ be the function such that $g(S) = f(V \setminus S)$ for all $S \subseteq V$. Then g is also submodular.*

Proof. Consider any two sets A and B. Using the submodularity of f, we have

$$\begin{aligned} g(A) + g(B) &= f(V \setminus A) + f(V \setminus B) \\ &\geq f((V \setminus A) \cap (V \setminus B)) + f((V \setminus A) \cup (V \setminus B)) \\ &= f(V \setminus (A \cup B)) + f(V \setminus (A \cap B)) \\ &= g(A \cup B) + g(A \cap B). \end{aligned}$$

◀

▶ **Proposition 16.** *Let $f_1, f_2 : 2^V \to \mathbb{R}$ be submodular functions. Then $f(S) = f_1(S) + f_2(V \setminus S)$ is also submodular.*

Proof. By Proposition 15, $g_2(S) = f_2(V \setminus S)$ is a submodular function. Hence, $f(S) = f_1(S) + g_2(S)$ is also submodular. ◀

▶ **Proposition 17.** *Let $f : 2^V \to \mathbb{R}$ be a submodular function. Let f' be the following function: $f'(S) = f(S) + f(V \setminus S)$ for each set $S \subseteq V$. Then f' is submodular and symmetric.*

Proof. We have $f'(V \setminus S) = f(V \setminus S) + f(S) = f'(S)$ so f' is symmetric. By Proposition 15, f' is also submodular. ◀

B Hardness of Monotone-restricted MSCA

In this section, we show that Monotone-restricted MSCA is Set Cover hard even if the assignment cost functions g_i are modular and the separation cost function h is symmetric. This shows that the factor of $\log n$ in Theorem 11 is necessary. In fact, for the special case of Monotone-restricted MSCA where the separation cost function h is symmetric submodular, it was already known that an $O(\log n)$-approximation can be achieved [3].

▶ **Theorem 18.** *There is an approximation preserving reduction from the* Set Cover *problem to the special case of* Monotone-restricted MSCA *in which each assignment cost function g_i is modular and the separation cost function h is symmetric. Moreover, each function g_i satisfies $g_i(S) = \sum_{v \in S} c(v, i)$, where $c(v, i)$ is either zero or infinity. The function h satisfies $h(A) = 0$ if $A \in \{\emptyset, V\}$ and $h(A) = 1$ otherwise.*

▶ Remark. The function $h : 2^V \to \mathbb{R}$ that satisfies $h(A) = 0$ if $A \in \{\emptyset, V\}$ and $h(A) = 1$ otherwise, is the cut function of a hypergraph on the vertex set V that has a single hyperedge containing all the vertices. This function is known to be symmetric submodular, which is easy to verify directly as well.

Our reduction is based on the reduction of Svitkina and Tardos [24] for Monotone MSCA. Consider an instance of Set Cover consisting of a set $V = \{v_1, \ldots, v_n\}$ of n elements and a collection $\mathcal{S} = \{S_1, \ldots, S_k\}$ of k sets. We construct an instance of Monotone-restricted MSCA as follows. The ground set is the set V of elements in the Set Cover instance. We have a label i for each set S_i in $\mathcal{S}$. For each element v and each label i, we have an assignment cost $c(v, i)$ that is equal to zero if $v \in S_i$ and ∞ otherwise. The assignment cost function g_i for the i-th label is defined as follows: $g_i(A) = \sum_{v \in A} c(v, i)$ for each set $A \subseteq V$. The separation function h is defined as above: $h(A) = 0$ if $A \in \{\emptyset, V\}$ and $h(A) = 1$ otherwise.

Note that we may assume that there does not exist a set in $\mathcal{S}$ that covers all the elements, since otherwise the solution consisting of such a set is an optimal solution (and this does not happen in hard instances of Set Cover).

▶ **Lemma 19.** *Suppose that there does not exist a set in $\mathcal{S}$ that covers all the elements. Then the* Set Cover *instance has a solution consisting of t sets if and only if the* Monotone-restricted MSCA *instance has a solution of cost t.*

Proof. Consider a solution $\mathcal{S}' \subseteq \mathcal{S}$ for the Set Cover instance. We construct a labeling $A_1, \ldots, A_k$ inductively as follows. We let $A_1 = S_1$ if $S_1 \in \mathcal{S}'$ and $A_1 = \emptyset$ otherwise. Consider an index $i \geq 2$. We let $A_i = S_i \setminus (A_1 \cup \cdots \cup A_{i-1})$ if $S_i \in \mathcal{S}'$ and $A_i = \emptyset$ otherwise.

Note that the resulting sets $A_1, \ldots, A_k$ are disjoint and they cover all the elements. Since $A_i \subseteq S_i$, we have $c(v, i) = 0$ for each $v \in A_i$ and thus $g_i(A_i) = 0$. Additionally, $h(A_i) = 1$ only if $S_i \in \mathcal{S}'$. Therefore the total separation cost of the labeling is at most $|\mathcal{S}'|$.

Conversely, consider a solution $A_1, \ldots, A_k$ for the Monotone-restricted MSCA instance. Note that we may assume that the solution has finite cost and thus $g_i(A_i) = 0$ for all labels i. It follows that $A_i \subseteq S_i$ for each i. We construct a set cover $\mathcal{S}'$ as follows. For each i such that A_i is non-empty, we add S_i to $\mathcal{S}'$. Since the sets A_i cover all the elements and $A_i \subseteq S_i$

for each i, the sets of $\mathcal{S}'$ cover all the elements as well. Since $V \notin \mathcal{S}$, $A_i \neq V$ for all i. Thus the cost of the labeling is equal to the number of non-empty sets in the labeling, which in turn it is equal to $|\mathcal{S}'|$. ◀